Stanford Machine Learning W6 Advice.

1. Deciding What To do Next.

- Debugging → error ↑
    - larger training data.
    - smaller set of feature.
    - get additional features.
    - polynomial features.
    - ↑ λ or ↓ λ

  how to pick one of the options?

- Machine Learning Diagnostic.

2. Evaluating a Hypothesis

(1) split data set (randomly sorted) → Training set 0.7, Test set 0.3

m_{test} = # of test examples,

(2) Procedure

(1) learn θ s.t. $\min_\theta J(\theta)$

(2) lin. reg. $J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} [h_\theta(x^{(i)}_{test}) - y^{(i)}_{test}]^2$ (linear reg.)

log. reg $J_{test}(\theta) = \frac{-1}{m_{test}} \sum_{i=1}^{m_{test}} y^{(i)}_{test} \cdot \log h_\theta(x^{(i)}_{test}) + (1-y^{(i)}_{test}) \log h_\theta(x^{(i)}_{test})$

misclassification error.

$$err(h_\theta(x), y) = \begin{cases} 1 & (h_\theta(x) \geq 0.5 \ \& \ y=0) \ || \ (h_\theta(x) \leq 0.5 \ \& \ y=1) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Test error} = \frac{1}{m_{test}} \sum_{i=1}^{m_{test}} err(h_\theta(x^{(i)}_{test}), y^{(i)}_{test})$$

3. Model Selection and Train / Validation / Test set

- Overfitting : cost < generalization error.

- Model selection (d: $\deg_x(h)$)

$d=1$ $\quad h = \theta_0 + \theta_1 x \quad \rightarrow \theta^{(1)} \rightarrow J_{test}(\theta^{(1)})$

$d=2$ $\quad h = \theta_0 + \theta_1 x + \theta_2 x^2 \quad \rightarrow \theta^{(2)}$

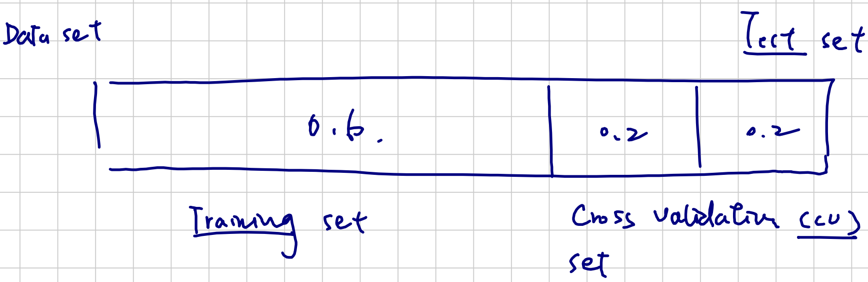
$\vdots \qquad \vdots \qquad \vdots$

$d=10$ $\quad h = \theta_0 + \theta_1 x + \cdots + \theta_{10} x^{10}. \quad \rightarrow \theta^{(10)} \quad J_{test}(\theta^{(10)})$

- problem. $J_{test}(\theta)$ is likely to be an optimistic generalization error.

  (our extra parameter d is fit to test)

$\downarrow$



Train / validation / test error.

Training error. $$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x^{(i)}) - y^{(i)}]^2$$

Cross validation error. $$J_{train}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} [h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)}]^2$$

Test error $$J_{test}(\theta) = \frac{1}{2m_{test}} \sum_{i=1}^{m_{test}} [h_\theta(x_{test}^{(i)}) - y_{test}^{(i)}]^2$$

Procedure

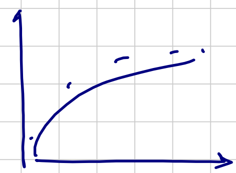
get $\theta^{(d)}$ → test on cv set. $J_{cv}(\theta^{(d)})$ $\forall d$.

→ find d s.t. $J_{cv}(\theta^{(d)})$ is min.

→ estimate generalization error for test set $J_{test}(\theta^{(4)})$

4. Diagnosing Bias v.s. Variance.

high bias
underfit



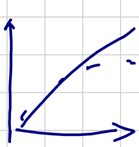
high variance
overfit

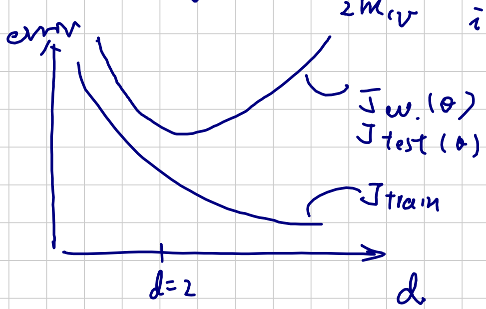
Bias / Variance

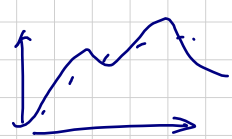
$$J_{train}(\theta) = \frac{1}{2m}\sum_{i=1}^{m}[h_\theta(x^{(i)}) - y^{(i)}]^2$$

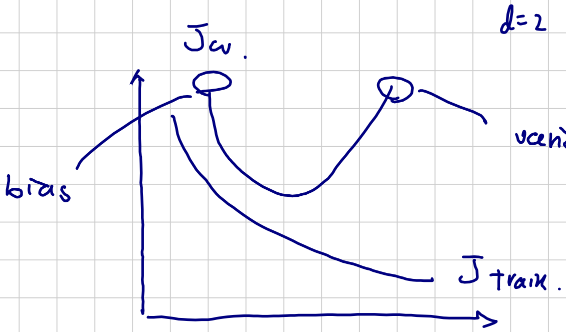
(or $J_{test}(\theta)$)

Cross validation error $$J_{cv}(\theta) = \frac{1}{2m_{cv}}\sum_{i=1}^{m_{cv}}[h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)}]^2$$









variance (overfit, d too large).

Bias [underfit] → $J_{train}(\theta)$ high
$J_{cv}(\theta) \approx J_{train}(\theta)$

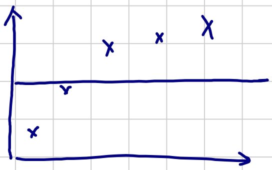
Variance [overfit] → $J_{train}(\theta)$ low
$J_{cv}(\theta) >> J_{train}(\theta)$

5. Regularization and Bias / variance.

Linear regression with regularization.

Model: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_4 x^4$

$$J(\theta) = \underbrace{\frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x^{(i)}) - y^{(i)}]^2}_{J_{train}\ (w/o\ reg.)} + \underbrace{\frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2}_{reg.}$$

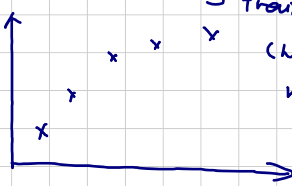


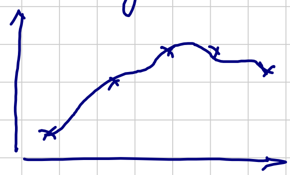
large λ.

$\lambda = 10000$, $\theta_{1,2,3,4} \approx 0$.

$h_\theta \simeq \theta_0$

high bias (under fit)

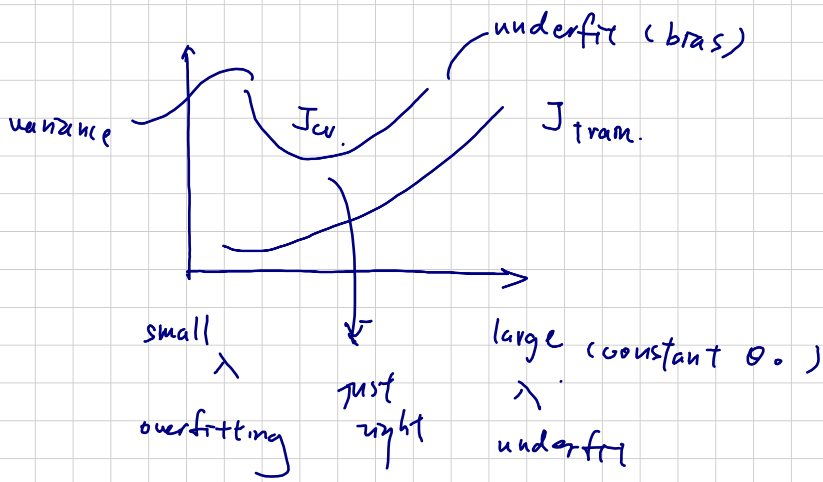




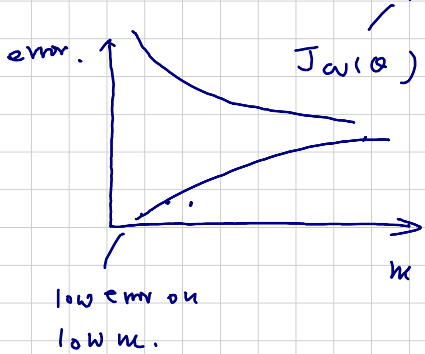
$\lambda = 0$ (small λ)

high variance (overfit)

Training error. (J_{train}) vs. validation error (J_{cv})



6. Learning Curve.

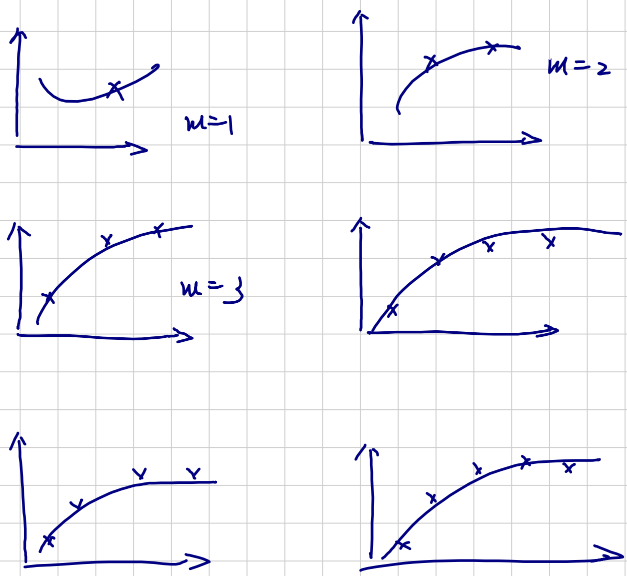


$$J_{train}(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x^{(i)}) - y^{(i)}]^2$$

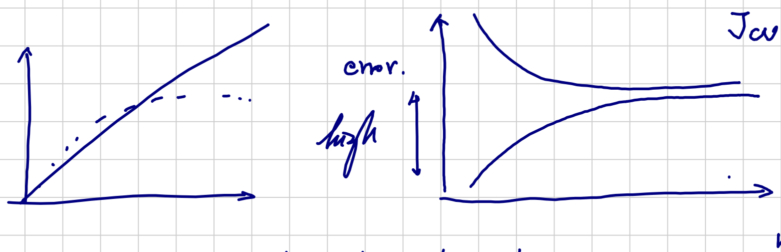
$$J_{cv}(\theta) = \frac{1}{2m_{cv}} \sum_{i=1}^{m_{cv}} [h_\theta(x_{cv}^{(i)}) - y_{cv}^{(i)}]^2$$

$\Rightarrow$ error $\sim$ m. "easier to fit in smaller training set"

for example $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2$



<u>High Bias</u>



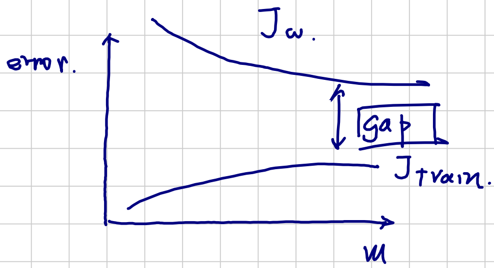
$J_{cv}(\theta)$ plateaus out.

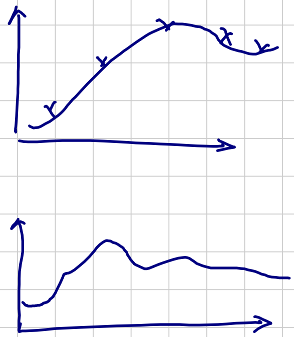
$J_{train}(\theta) \approx J_{cv}$. (small data set)

If learning algorithm has high bias,

↑ data set does not help.

High Variance





deg h ↑
small λ.

If a learning algorithm is suffering from high variance, getting more training data is likely to help.

7. Deciding What to Do Next Revisited.

- get more examples : fix high variance.
- try smaller set of feature : fix high variance

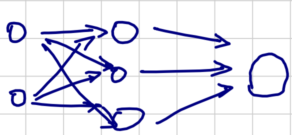
> over fitting.

- Add features : fix high bias.
- polynomial features : fix high bias

- try decr. λ ⟶ fix high bias (incr. importance of θ).
- try incr λ ⟶ fix high variance (decr. importance of θ).

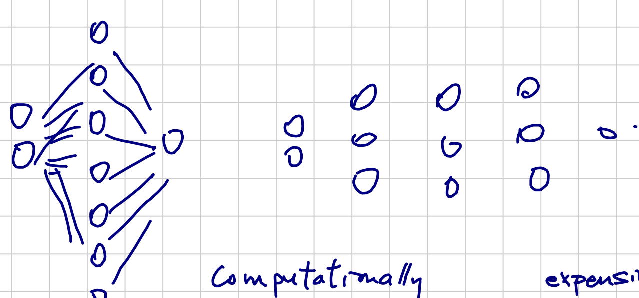
Neural Network and Overfitting

"Small" Neural network
(few parameters ⟶ prone to underfitting)



computationally cheap.

"Large" Neural Network
(more parameters ⟶ prone to overfitting)



how many hidden layers?

Computationally expensive.
→ use regularization (λ) to address overfitting.

Machine Learning System Design

1. Building a spam classifier

- Supervised learning $\quad x$ = features of email.

  $y = 1$ (spam) or 0 (not)

- features (x). → indicative words.

$$X = \begin{pmatrix} 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix} \begin{matrix} \text{buy} \\ \text{discount} \\ \vdots \end{matrix}$$

presence of word.

- Reduce error.
  - acquire more data.
  - develop sophisticated features based on email routing information (from email header)
  - " for message body.
  - develop sophisticated algorithm to detect misspellings e.g. w4tches

2. Error analysis.

- Get something "quick and dirty" → plot learning curve (start w/ simple learning algorithm). → decide next steps

- Error analysis → manually look at examples and try to elicit systematic errors.

Ex. m_{cv} = 500 examples. Algorithm miss-classifies 100 emails.

↳ manually examine.

(i) what type of miss-classification

(ii) what cues (features) you think would have helped the algorithm classify them correctly.

Importance of numerical evaluation

↳ single number to report how well the algorithm is performing.

- stemming software. (discount / discounts / discounted) ...

  (e.g. Porter spammer)

Skewed Data

- 1% error. → only 0.5% patients have cancer.
- skewed classes → one class is far more ubiquitous
- problem
  - e.g. 99.2% accuracy (0.8% error)

    ↓

    99.5% accuracy (0.5% error)

for example, function y = predict Cancer (x)

y=0;

return

→ good accuracy b/c of dataset

1. Precision / Recall .

prediction	Actual class 1	0
1	True positive	false positive
0	false negative	True negative

(1) Precision (Of all patients where we predicted $y=1$, what fraction actually has cancer ?)

$$= \frac{\text{True positive}}{\text{\# positive predictions}} = \frac{\text{true positives}}{(\text{true} + \text{false})\ \text{positives}}$$

(2) Recall (of all patients that have cancer, what fraction did we predict so ?)

$$= \frac{\text{true positive}}{\text{\# actual positives}} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}}$$

· Convention: $y=1$ in presence of rare class

2. Trading off Precision and Recall.

$$\begin{cases} \text{precision} = \dfrac{\text{true pos.}}{\text{pred. pos}} \\ \\ \text{recall} = \dfrac{\text{true pos}}{\text{actual pos.}} \end{cases}$$

· logistic regression: $0 \le h_\theta(x) \le 1$

threshold 0.5 ⟶ 0.7 (Incr. confidence).

$y=1$, $h_\theta(x) > 0.5$ $h_\theta(x) > 0.7$

↳ higher precision ⇒ more confident predictions

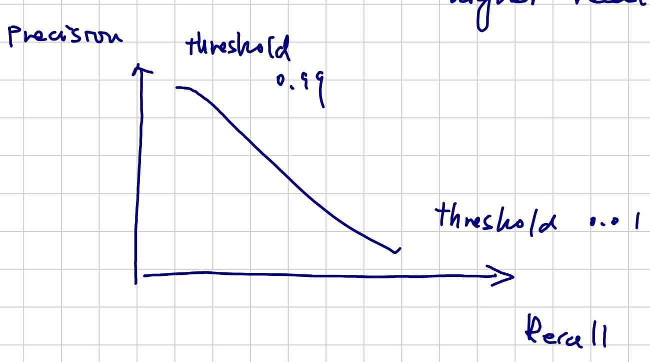
lower recall ⇒ conservative

threshold 0.5 ⟶ 0.3 [Avoid false negative]

when in doubt, predict $y=1$.

lower precision → less confident in our prediction.

higher recall. → less conservative



How to select threshold?

- scalar that reports how well our algorithm performs.

  ↳ Now we have 2! (precision / recall)

  ↓ need a single number.

(1) Average. $\frac{P+R}{2}$?

- caveat:

(predict)

$y=1 \gg y=0$ high recall.

$y=0 \gg y=1$ high precision.

> high skew our average

(2) F score.

$F_1 = \frac{2PR}{P+R}$; $P=0$ or $R=0 \rightarrow F_1=0$.

$P=1$ and $R=1 \rightarrow F_1 = 1$ (perfect precision and recall).

Data for Machine Learning

- Algorithms:
  - perceptron.
  - Winnow.
  - memory-based
  - naive bayes.

Large Data rationale

- Assume feature $x \in \mathbb{R}^{n+1}$ has sufficient information to predict y accurately.
- useful test: given the input x, can a human expert confidently predict y?

- Use learning algorithm with many parameters
  e.g. logistic regression / linear regression with many features
  neural network with many hidden units.

  ↳ low bias algorithms. $J_{train}(\theta)$ will be small

- Use large training set (unlikely overfit).
  $J_{train} \approx J_{test}$

⇒ J_{test} will be small